**PRACTICAL NO 4:**

**Aim:** **To implement the Dijstra Algorithm using backtracking approach.**

**Code:**

#include<stdio.h>

#include<conio.h>

#define INFINITY 9999

#define MAX 100

int G[MAX][MAX];

void dijkstra(int G[MAX][MAX],int n,int startnode);

int main()

{

int G[MAX][MAX],i,j,n,u,v1,v2,m,w;

printf("Enter no. of vertices:");

scanf("%d",&n);

printf("Enter no. of edges:");

scanf("%d",&m);

printf("Enter the connention of the vertices and Weight\n");

printf("vertex1\tvertex2\tweight\n");

for(int i=0; i<m; i++)

{

scanf("%d %d %d",&v1,&v2,&w);

//direted

G[v1][v2]=w;

G[v2][v1]=w;

}

printf("\n The adjacency matrix:\n");

for(int i=0; i<n; i++){

for(int j=0; j<n; j++){

printf("%d ",G[i][j]);

}

printf("\n");

}

printf("\nEnter the starting node:");

scanf("%d",&u);

dijkstra(G,n,u);

return 0;

}

void dijkstra(int G[MAX][MAX],int n,int startnode)

{

int cost[MAX][MAX],distance[MAX],pred[MAX];

int visited[MAX],count,mindistance,nextnode,i,j;

for(i=0;i<n;i++)

for(j=0;j<n;j++)

if(G[i][j]==0)

cost[i][j]=INFINITY;

else

cost[i][j]=G[i][j];

for(i=0;i<n;i++)

{

distance[i]=cost[startnode][i];

pred[i]=startnode;

visited[i]=0;

}

distance[startnode]=0;

visited[startnode]=1;

count=1;

while(count<n-1)

{

mindistance=INFINITY;

for(i=0;i<n;i++)

if(distance[i]<mindistance&&!visited[i])

{

mindistance=distance[i];

nextnode=i;

}

visited[nextnode]=1;

for(i=0;i<n;i++)

if(!visited[i])

if(mindistance+cost[nextnode][i]<distance[i])

{

distance[i]=mindistance+cost[nextnode][i];

pred[i]=nextnode;

}

count++;

}

for(i=0;i<n;i++)

if(i!=startnode)

{

printf("\nDistance of node%d=%d",i,distance[i]);

printf("\nPath=%d",i);

j=i;

do

{

j=pred[j];

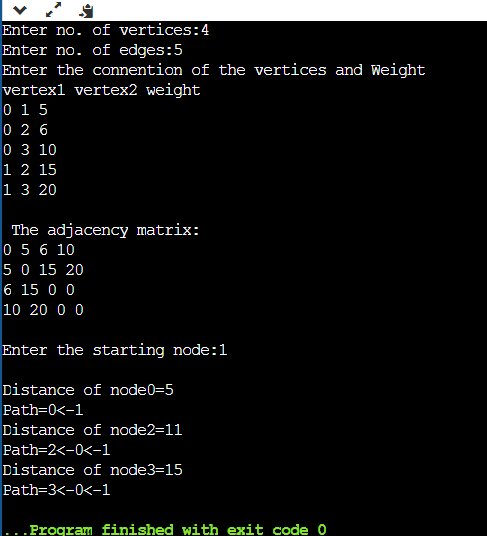
printf("<-%d",j);

}while(j!=startnode);

}

}

**Output:**



**Conclusion: Dijkstra Algorithm using backtracking approach was Implemented successfully.**